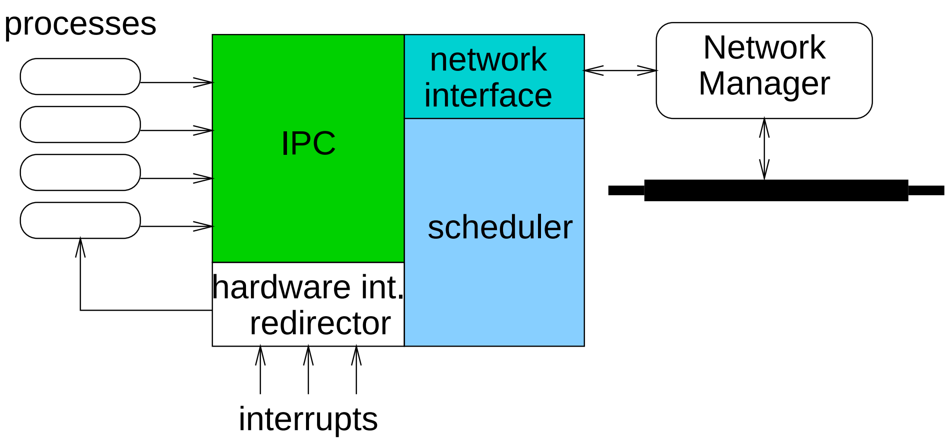
Teaching material for RTOS, QNX Operating System

Dániel Drótos

October 27, 1997

QNX microkernel operating system



QNX Microkernel Functions

1. Interpocess communication

   Messages, Proxies, Signals

2. Low-level network communication

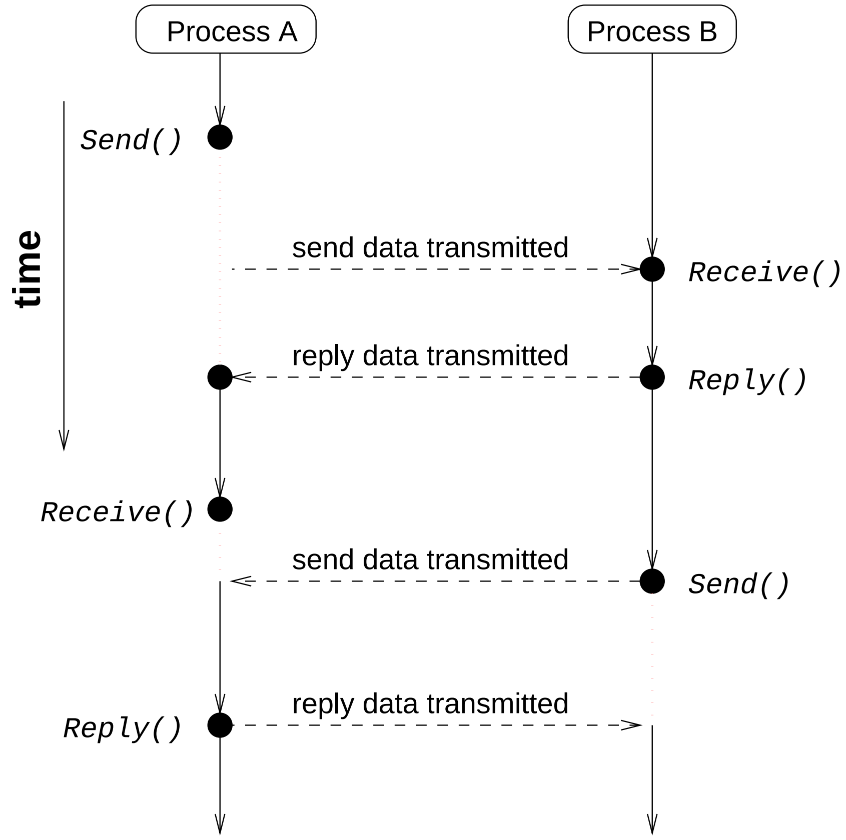
3. Process scheduling

4. First-level interrupt handling

IPC via messages

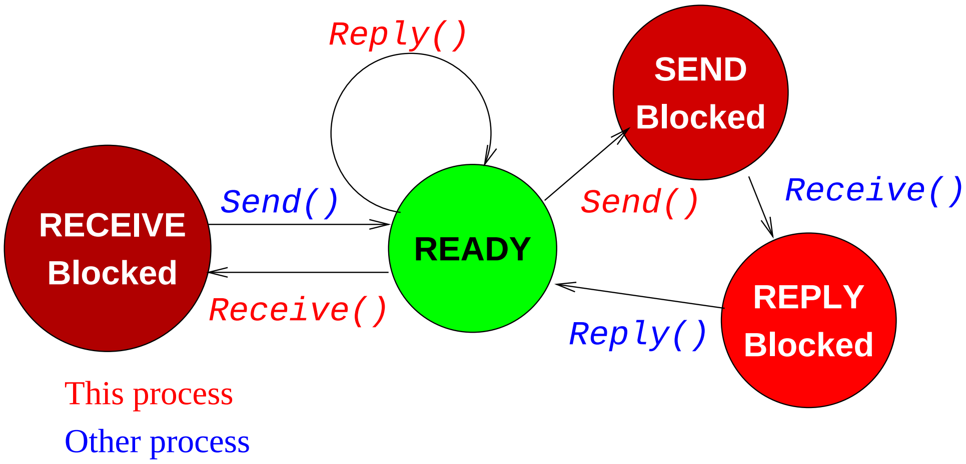
Functions:

Send()
Receive()
Reply()



IPC via messages

Blocked states



IPC via messages

Advanced facilities

Conditional message reception

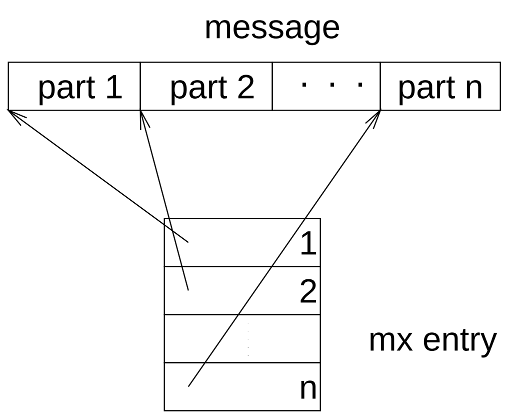
Creceive()

Reading or writing part of a message

Readmsg(), Writemsg()

Multi part messages

Creceivemx(), Readmsgmx(),
Receivemx(), Sendmx(), Writemsgmx()



IPC via proxies

Proxy is a non-blocking, predefined message

Using proxies:

1. Creating

qnx_proxy_attach()

2. Sending

Trigger()

Proxies can be sent more than once by any process who knows its identification.

3. Receiving

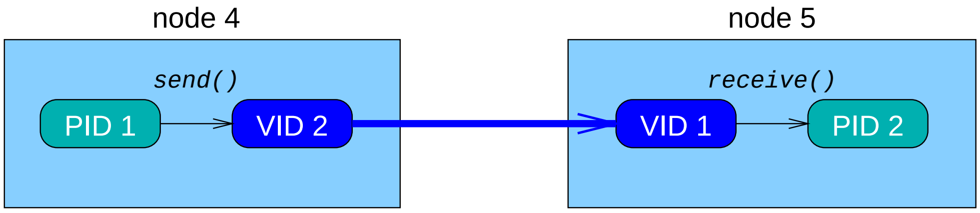
Receive()

Proxies can be received just like any other message.

IPC across the network

At the application level, there is no difference between a local and a remote resource.

Transparency is made by *virtual circuits*



VID of VC holds:
- local pid
- remote pid
- remote nid
- remote vid

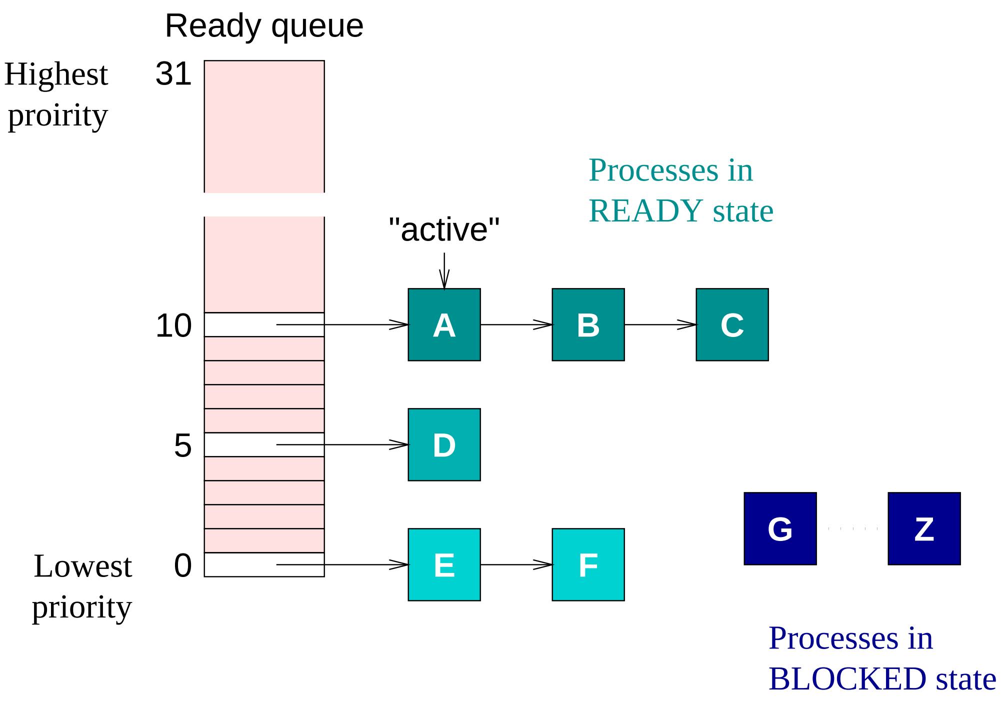
VCs are created

- by *qnx_vc_attach()*
- automatically by *open(), qnx_name_locate(), etc.*

Process scheduling

The Kernel's scheduler makes decisions when:

- a process becomes unblocked
- the timeslice for a running process expires
- a running process is preempted



Scheduling methods

FIFO scheduling a process continues executing until:

- voluntarily relinquishes (e.g. it blocks)
- is preempted by a higher-priority process

Round-robin scheduling a process continues executing until:

- voluntarily relinquishes control
- is preempted by a higher-priority process
- consumes its timeslice

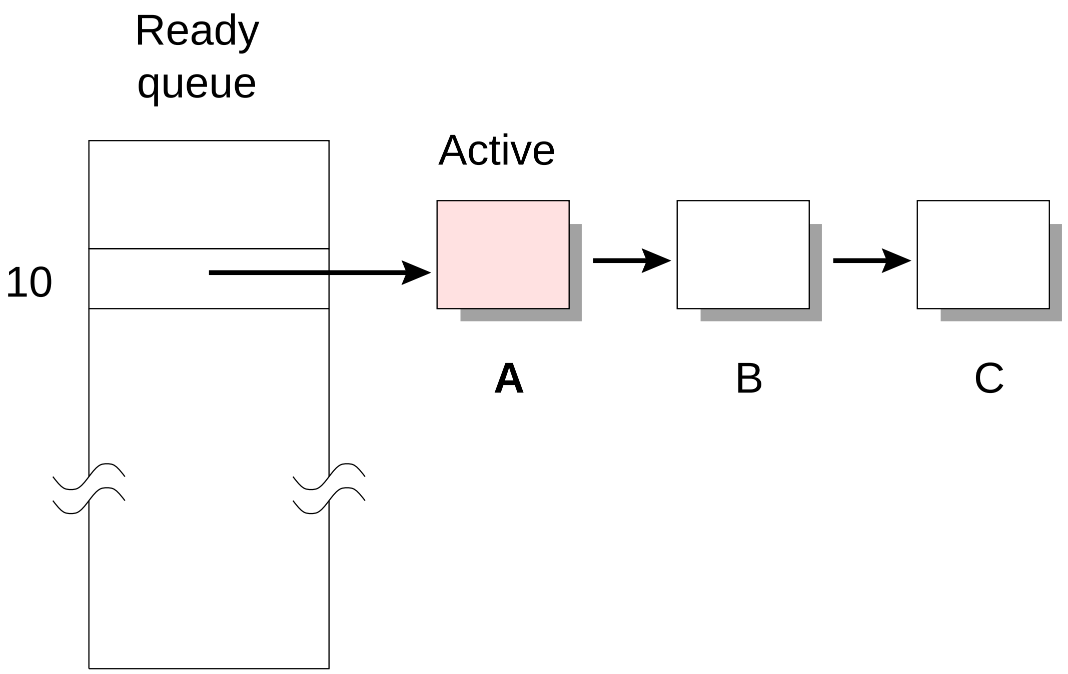
Adaptive scheduling

- if the process consumes its timeslice, its priority is reduced by 1 if another process at the same priority is READY (this known as priority decay)

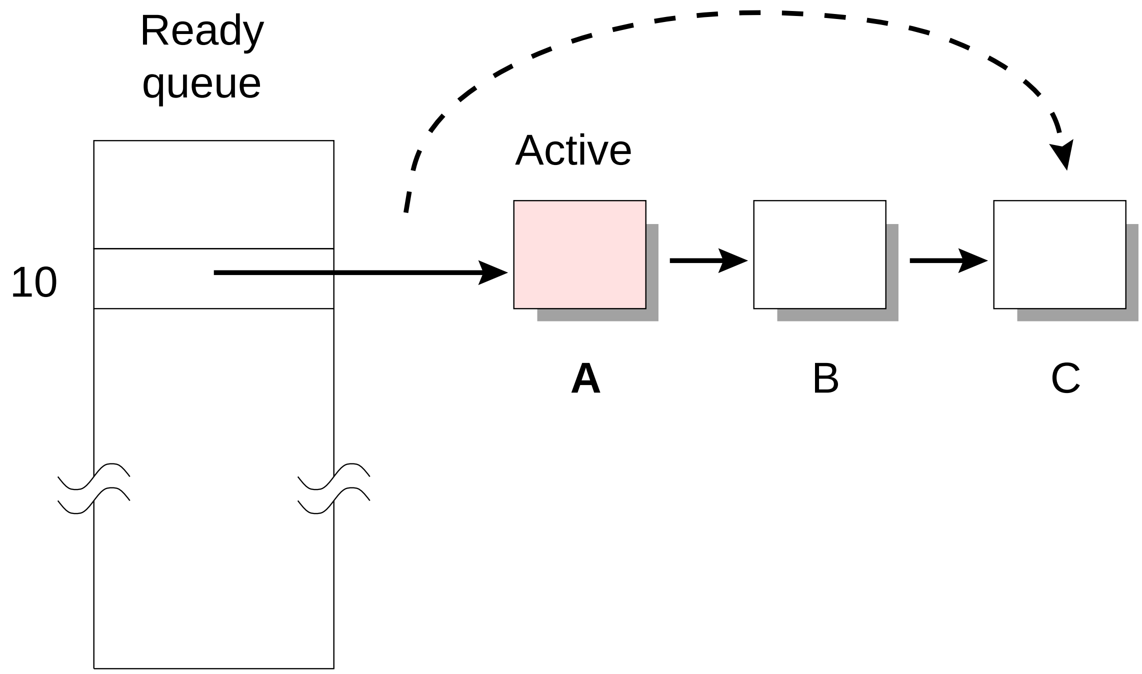
- if the process has decayed and remains unscheduled for a second, its priority is boosted by 1

- if the process blocks, it immediately reverts to its original priority

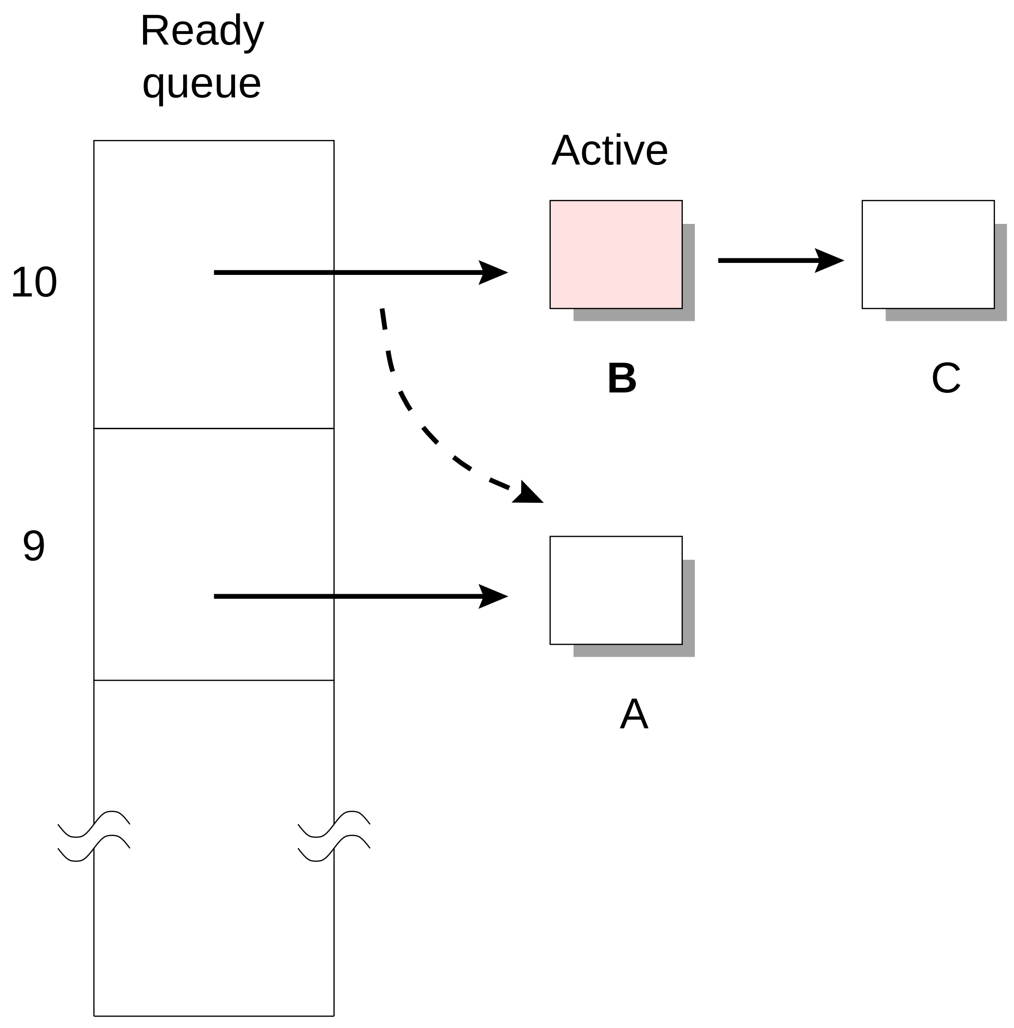
FIFO Scheduling Method



Round-Robin Scheduling Method



Adaptive Scheduling Method



Process management

1. Process creation

fork()
exec()
spawn()

2. Process inheritance

3. Process termination

signal

exit() function call

Termination involves two stages

1. Termination thread in process manager is run
2. Notification of process termination is sent to the parent process

Process states

READY

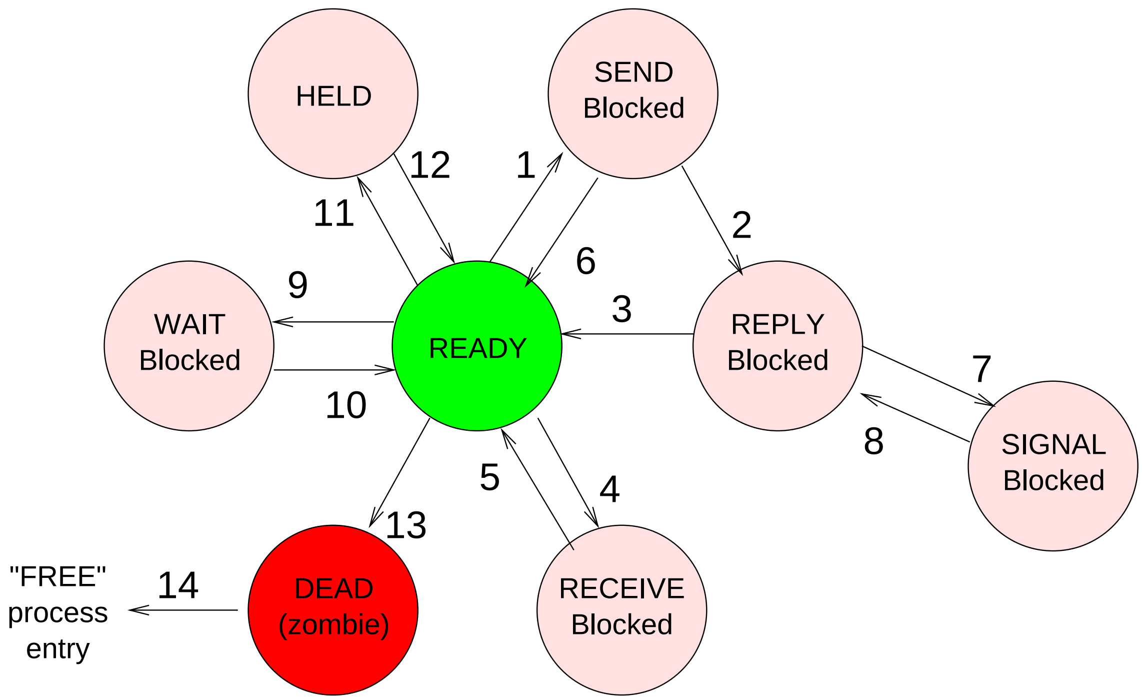
BLOCKED

- SEND-blocked
- RECEIVE-blocked
- REPLY-blocked
- SIGNAL-blocked

HELD

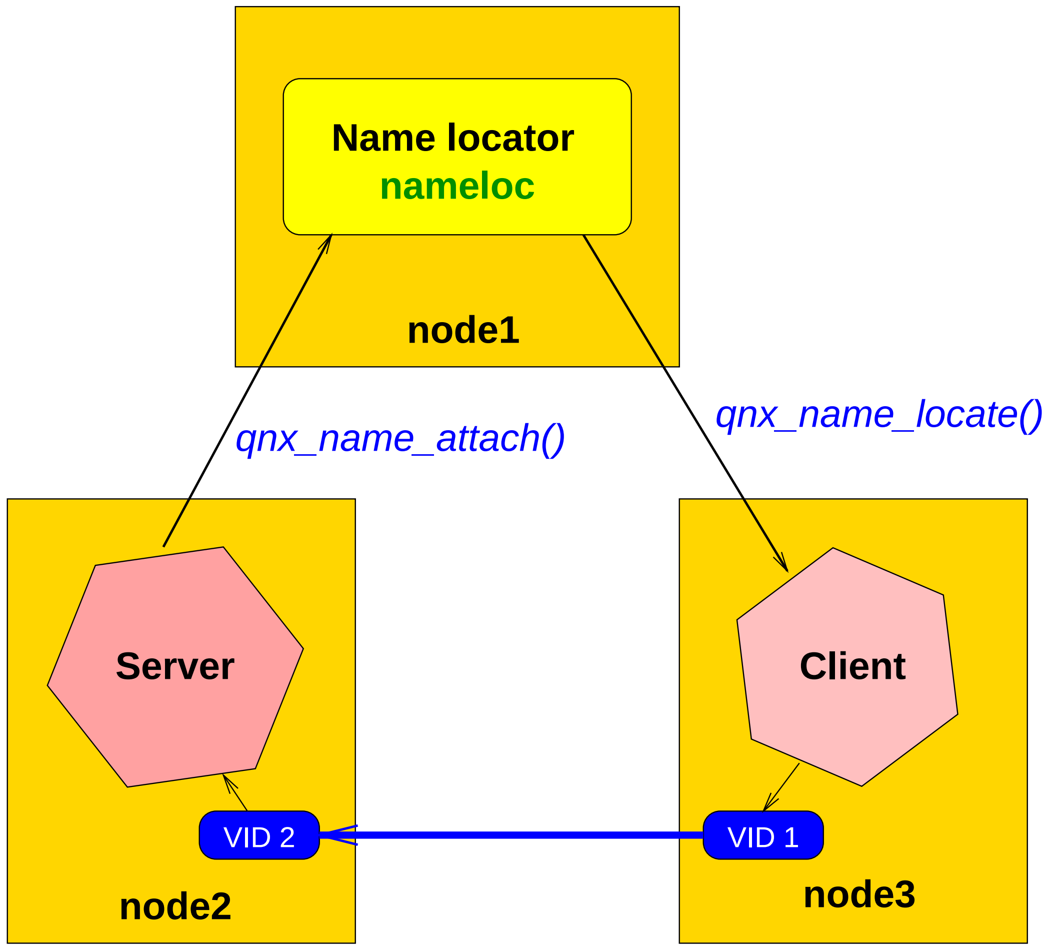
WAIT-blocked

DEAD



Process symbolic names

Applications can be split into cooperating processes
e.g. client-server model



qnx_name_locate() makes virtual channels

Timers

1. Make timer

timer_create()

- sleep until completion
- notify with proxy
- notify with signal

2. Start/stop timer

timer_settime()

3. Inspect timer

timer_gettime()

4. Remove timer

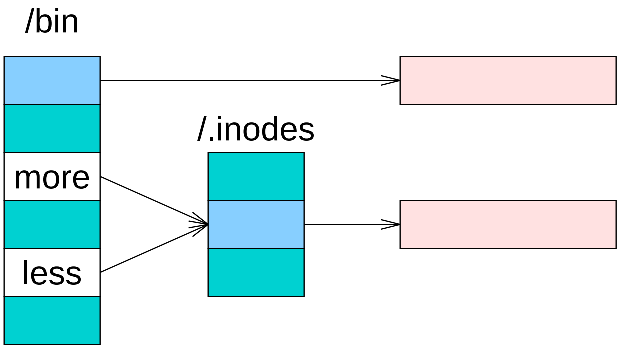
timer_delete()

Filesystem manager

Fsys maintains four different times for each file:

1. date of last access (read)
2. date of last write
3. date of last modification
4. date of creation

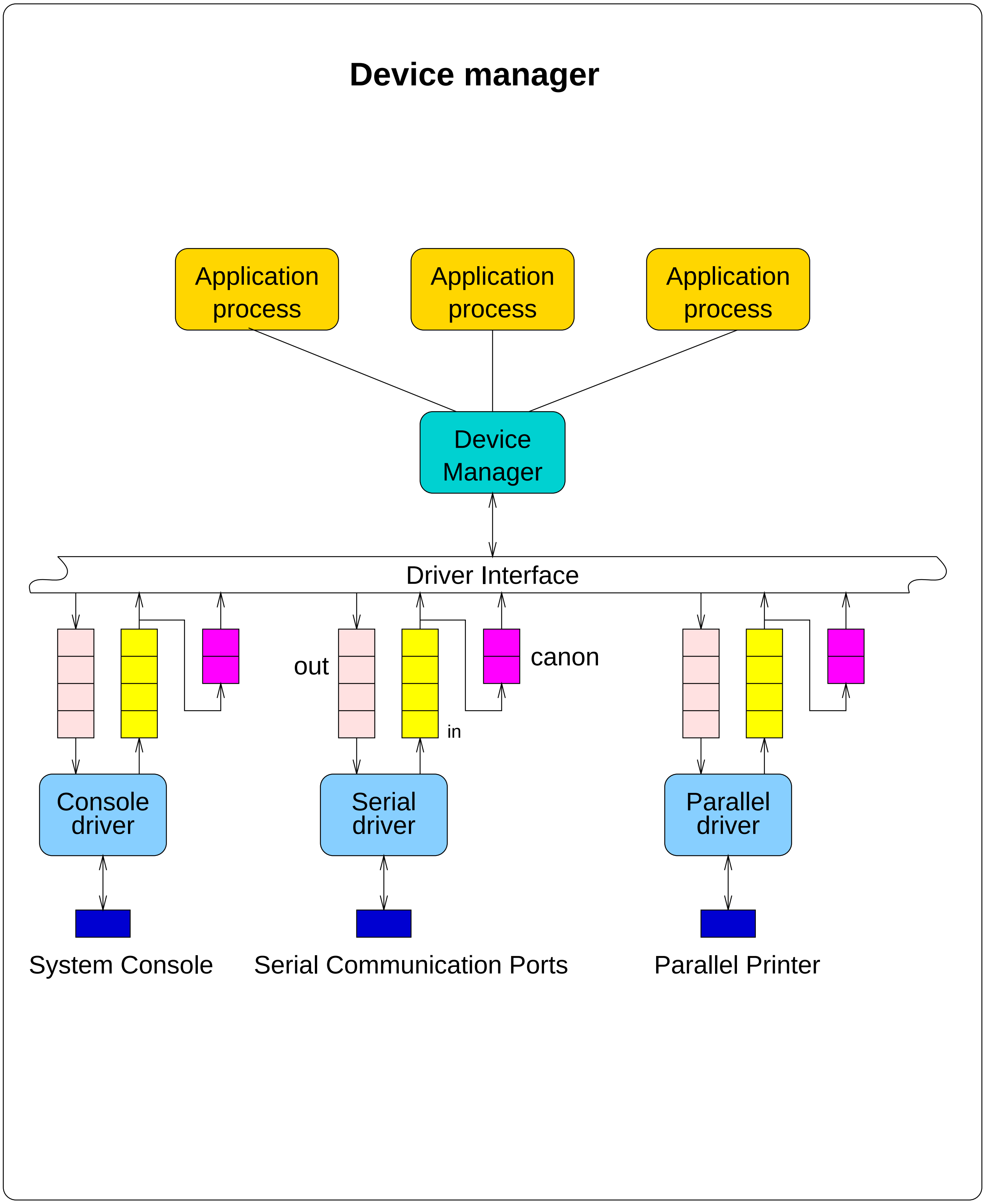
Regular files
Directories
Links and inodes

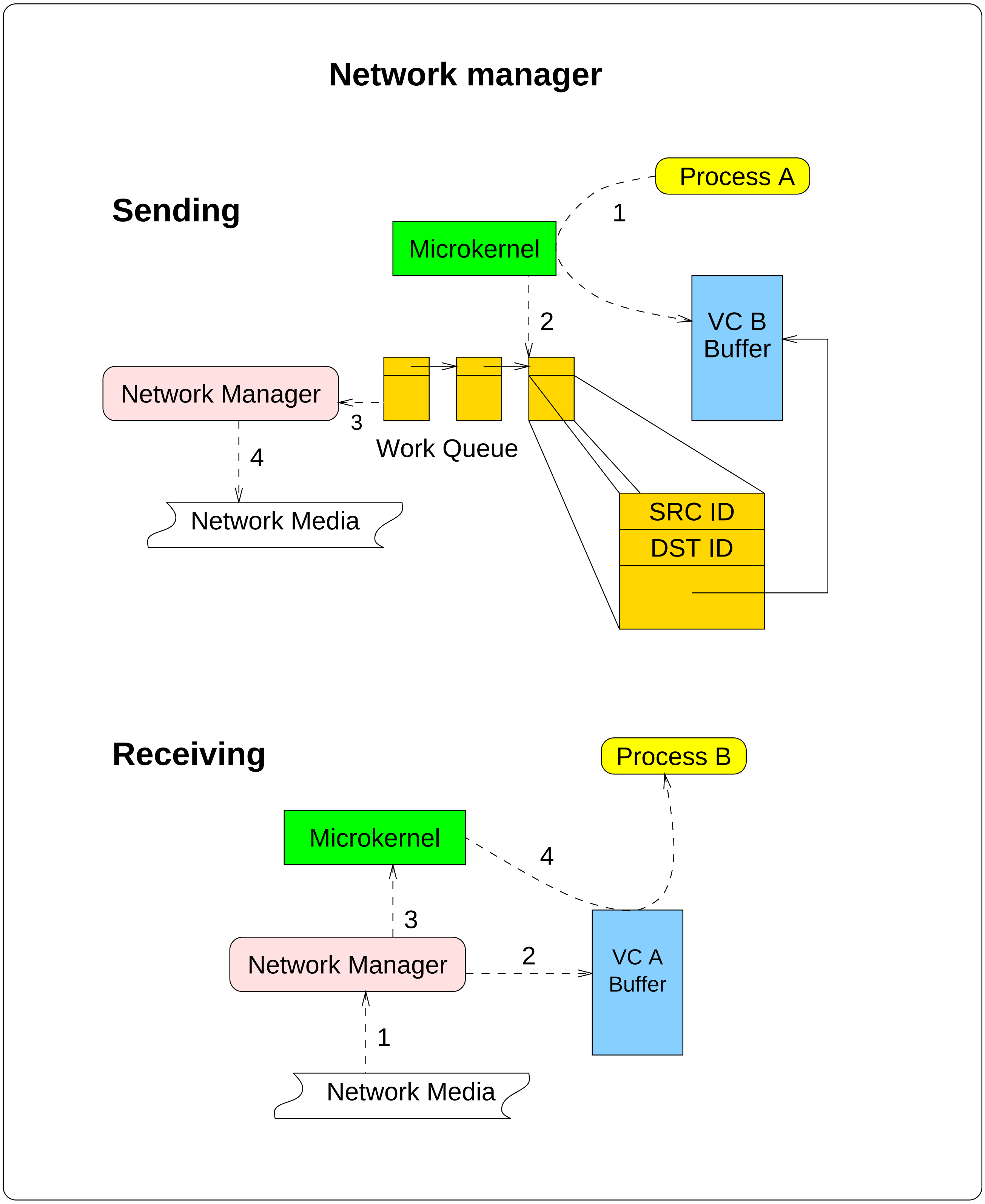


Symbolic links

Pipes

FIFOs


Device manager
Application process
Application process
Application process
Device Manager
Driver Interface
out
canon
in
Console driver
Serial driver
Parallel driver
System Console
Serial Communication Ports
Parallel Printer


Network manager
Sending
Process A
1
Microkernel
VC B
Buffer
2
Network Manager
3
Work Queue
4
Network Media
SRC ID
DST ID
Receiving
Process B
Microkernel
4
3
Network Manager
2
VC A
Buffer
1
Network Media